UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/690,343	10/20/2003	Panagiotis Kougiouris	11289-036-999 (902782-999	4229

20583 7590 08/03/2005

JONES DAY
222 EAST 41ST ST
NEW YORK, NY 10017

EXAMINER	
LIN, WEN TAI	

ART UNIT	PAPER NUMBER
2154	

DATE MAILED: 08/03/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/690,343	KOUGIOURIS ET AL.	
	Examiner	Art Unit	
	Wen-Tai Lin	2154	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *20 October 2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-41* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-41* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *10/8/04* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *3/7/05*, *10/20/03*
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-41 are presented for examination.

Claim Rejections - 35 USC § 102

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

3. Claims 1-5, 12-13, 15, 18, 22-25, 33, 35 and 38 are rejected under 35 U.S.C. 102(e) as being anticipated by Brown[U.S. Pat. No. 5857190].

4. As to claim 1, Brown teaches the invention as claimed including: a system for logging an event, comprising:

a computer system including a CPU and memory[Fig.1];

a server-side logging component comprised in the memory of the computer system [56; Fig.1];

a means for persistently storing information[e.g., 62, 64, Fig.1];

a first client-side logging component comprised in the memory of the computer system [36, Fig.1; col.7, lines 12-14];

a first executable module [e.g., 60, Fig.1; col.4, lines 50-65; that is, each service application may be viewed as an executable module], wherein the first executable module is executable to call the first client-side logging component in order to log an event, wherein the event comprises an event level[200, Fig. 4; col.6, lines 39-58; 80, Fig.3; that is, any service application associated with a respective interrupt handling routine constitutes the first executable module, causing the event evaluator to log an event, wherein the event evaluator is implemented as an API-Object];

wherein the first executable module calling the first client-side logging component comprises the first executable module passing information specifying the event level [col.6, lines 26-33]to the first client-side logging component [col.17, lines 8-13; Table 1; i.e., since the event level is part of the information that the event evaluator (log-API) reports to the server, it is obvious that the same information must have been passed along from the event capture (200, Fig.4) associated with the first executable module];

wherein the first client-side logging component calling the server-side logging component comprises the first client-side logging component passing information specifying the event level to the server-side logging component[col.17, lines 8-13];

wherein the server-side logging component persistently logging the event comprises the server-side logging component persistently logging the event level[82, 90, 88, Fig.3];

wherein, in response to said first executable module calling the first client-side logging component in order to log an event, the first client-side logging component is executable to call

the server-side logging component in order to log the event[80, 84, 82, Fig.3; col.14, lines 37-38; col.2, lines 36-43]; and

wherein in response to said first client-side logging component calling the server-side logging component in order to log the event, the server-side logging component is executable to utilize the means for persistently storing information in order to persistently log the event [col.10, line 66 - col.11, line 11; abstract, lines 12-13].

5. As to claim 2, Brown further teaches that the first client-side logging component maintains logging criteria information specifying which levels of events should be logged[54, Fig.3; col.12, lines 40-67];

wherein, in response to said first executable module calling the first client-side logging component, the first client-side logging component is executable to compare the information specifying the event level to the logging criteria information, in order to determine whether the event should be logged and wherein, if the first client-side logging component determines that the event should not be logged, the client-side logging component does not perform said calling the server-side logging component [col.10, lines 4-9; 202, 204, 206, 210, Fig.4].

6. As to claim 3, Brown further teaches that the event is associated with a first event category [col.5, lines 27-28 and 65-67];

wherein said first executable module calling the first client-side logging component comprises the first executable module passing information specifying the first event category to the first client-side logging component[200, Fig. 4; col.6, lines 39-58; 80, Fig.3; that is, any

service application associated with a respective interrupt handling routine constitutes the first executable module, causing the event evaluator to log an event, wherein the event evaluator is implemented as an API-Object];

wherein said first client-side logging component calling the server-side logging component comprises the first client-side logging component passing information specifying the first event category to the server-side logging component[col.17, lines 8-13];

wherein said server-side logging component persistently logging the event comprises the server-side logging component persistently logging the first event category[82, 90, 88, Fig.3].

7. As to claim 4, Brown further teaches that the first client-side logging component maintains logging criteria information specifying which categories of events should be logged [col.18, line 60 - col.19, line 2; that is, the logging criteria may be updated by the event evaluator in response to a system server issuing the change (col.18, lines 43-59)];

wherein, in response to said first executable module calling the first client-side logging component, the first client-side logging component is executable to compare the information specifying the first event category to the logging criteria information, in order to determine whether the event should be logged[202, 204, Fig.4];

wherein, if the first client-side logging component determines that the event should not be logged, the client-side logging component does not perform said calling the server-side logging component [col.15, line 60-col.16, line 5].

8. As to claim 5, Brown further teaches that the event is also associated with a second event category [col.6, lines 13-14; Table 3];

wherein said first executable module calling the first client-side logging component further comprises the first executable module passing information specifying the second event category to the first client-side logging component [col.17, lines 8-13; Table 1; i.e., since the event level is part of the information that the event evaluator (log-API) reports to the server, it is obvious that the same information must have been passed along from the event capture (200, Fig.4) associated with the first executable module]; and

wherein, in response to said first executable module calling the first client-side logging component, the first client-side logging component is executable to compare the information specifying the second event category to the logging criteria information, in order to determine whether the event should be logged [202, 204, Fig.4].

9. As to claim 12, Brown teaches that the system further comprising:

a logging administration tool executable to configure the server-side logging component with event-logging criteria information, wherein the event-logging criteria information includes information specifying which levels of events should be logged [92, Fig.6; col.12, lines 10-17; Table 3];

wherein the first client-side logging component maintains logging criteria information specifying which levels of events should be logged [col.12, lines 24-29];

wherein the server-side logging component is executable to propagate event-logging criteria information to the first client-side logging component in response to said logging

administration tool configuring the server-side logging component with event-logging criteria information [col.2, lines 49-52].

10. As to claim 13, Brown further teaches that the first client-side logging component is executable to update the logging criteria information that it maintains, in response to said server-side logging component propagating the event-logging criteria information to the first client-side logging component [col.19, lines 1-2].

11. As to claim 18, Brown further teaches that the means for persistently storing information comprises an element from the group consisting of a file and a database [62, 64, Fig.1; 88, Fig.3].

12. As to claims 15, 22-25, 33, 35 and 38, since the features of these claims can also be found in claims 1-5, 12-13 and 18, they are rejected for the same reasons set forth in the rejection of claims 1-5, 12-13 and 18 above.

Claim Rejections - 35 USC § 103

13. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

14. Claims 6-8, 26-32 and 39-41 are rejected under 35 U.S.C. 103(a) as being unpatentable over Brown [U.S. Pat. No. 5857190], as applied to claims 1-5, 12-13, 15, 18, 22-25, 33, 35 and 38 above, further in view of Goldsmith [U.S. Pat. No. 5491800].

15. As to claim 6, Brown does not specifically teach that the first client-side logging component (i.e., the LOG API) executes in-process with the first executable module.

However, Goldsmith teaches that an API object can be instantiated to reside in the same address space of a calling process. For example, communication using remote procedural call (RPC) can substantiate an API object on each side of the communcating nodes for interfacing to the calling modules [e.g., 718, 758, Fig.7; col.11, lines 8-17 and lines 46-58].

It would have been obvious to one of ordinary skill in the art at the time the invention was made that Brown's client-side logging component (i.e., LOG API) could have been instantiated to reside in the same address space of the first executable module (i.e., in-process with the executable module), so that communication between the first executable module and the client-side logging component can be streamlined (i.e., comparing to the case when the LOG API runs in the kernel space and the calling module runs in an application space).

16. As to claim 7, Brown in view of Goldsmith teach that the fist client-side logging component may run in a same process comprising one or multiple modules (see the reasons set forth for the rejection of claim 7 above).

Brown does not specifically teach that the system (e.g., service application program) comprises first and second executable modules wherein both modules are executable to call the first client-side logging component in order to log an event.

However, it is well known in the art that, based on modular programming style, a process can comprise a plurality of executable modules.

It would have been obvious to one of ordinary skill in the art at the time the invention was made that Brown's service application programs such as EPG and VOD (see col.4, lines 50-52) each could have been written in terms of multiple executable modules (e.g., one module takes care of user inputs from a remote controller, while another module handles user inputs directly from the TV panel), each having the capability of reporting events by calling the client-side logging component (i.e., an instantiated API-log object residing in the same address space) because a service application normally involves different types of user interface, and by doing so the overall application program is further modularized.

17. As to claim 8, Brown in view of Goldsmith further teaches that the first client-side logging component executes as an in-process COM object [Fig.2; col.9, lines 21-26].

18. As to claims 26-32 and 39-41, since the features of these claims can also be found in claims 1, 6-7, 9-11, 22 and 24-25, they are rejected for the same reasons set forth in the rejection of claims 1, 6-7, 9-11, 22 and 24-25 above.

19. Claims 9-11, 14, 16-17, 19-21, 34 and 36-37 are rejected under 35 U.S.C. 103(a) as being unpatentable over Brown [U.S. Pat. No. 5857190], as applied to claims 1-8, 12-13, 15, 18, 22-33, 35 and 38-41 above.

20. As to claim 9, Brown does not specifically teach that the system comprises two independent processes, each having an executable module and a client-side logging component.

However, Brown taught that the headend server is capable of providing different services to the clients [col.4, lines 47-55], wherein the activation of each service program naturally creates an independent process.

It would have been obvious to one of ordinary skill in the art at the time the invention was made that each of Brown's application process must have included at least an executable module and a client-side logging component so that the user's activities (such as press keys or changing channels) can be communicated to the server.

21. As to claim 10, Brown teaches a system for logging an event, substantially as claimed in claims 1-9 above.

Brown further teaches that that in response to said first executable module calling the first client-side logging component in order to log an event, the first client-side logging component is executable to store the event in a buffer [col.7, lines 59-67] and return execution control to the first executable module prior to evaluating the queued events [i.e., before the buffered events are reported periodically in batch mode, the execution control is naturally returned to the EPG

application (i.e., the evaluator) due to the fact that the event detection is performed in interrupt mode (col.6, lines 38-44)],

wherein said first client-side logging component calling the server-side logging component comprises the first client-side logging component asynchronously retrieving the event from the buffer and then calling the server-side logging components [col.7, lines 59-67; that is, it is obvious that the event transfer is performed asynchronously in particular when the transfer mode is based on a selected number of events having been accumulated in the buffer].

Brown does not specifically teach that the buffer is a queue. However, it is well known to implement an event buffer as a queue due to its capability to keep the order of the event. For the same reason, it is obvious to one or ordinary skill in the art to implement Brown's event buffer as a queue because Brown's event buffer also has to keep the events' occurring order.

22. As to claim 11, Brown does not specifically teach that the first client-side logging component asynchronously retrieving the event from the queue and then calling the server-side logging component is performed by an event queue manager thread.

However, multi-thread programming is a well-known concept in the art. Since the client-server event flow can be naturally partitioned into multiple stages (e.g. the first stage being a transfer between the executable module and the logging component and the second stage involving transfer between the logging component and the remote server), it would have been obvious to one of ordinary skill in the art to have constructed the client-side logging component task into multiple threads, so that each thread could be sequentially called in to complete the asynchronous transfer of event information.

23. As to claim 14, Brown further teaches that the server-side logging component is a COM component and the first client-side logging component is also a COM component [Fig.2; col.9, lines 21-26];

Brown does not specifically teach that the server-side logging component propagating the event-logging criteria information to the first client-side logging component is also performed by using COM mechanisms for sending a COM event from one COM component to another COM component.

However, Brown teaches that "by using a COM-based architecture, the logging service object can be moved to different servers at the headend as planning or organizational needs change, without affecting the event logging operation." [col.11, lines 42-45]. For this same reason, it is obvious that the propagation of the event-logging criteria information in Brown's system could also be performed by using COM mechanisms for sending a COM event from one COM component to another COM component.

24. As to claims 16-17, since the limitations of these claims can also be found in claims 1-6 and 10-16, they are rejected for the same reasons set forth in the rejection of claims 1-6 and 10-16 above.

As for the additional feature requiring time-stamping the event in claims 17 and 18: it is noted that such feature is well known in the art. It is obvious to one of ordinary skill in the art to time-stamp the events and pass it along to the server so that the server of Brown's system knows how to provide services to the clients in accordance with their requests in an orderly manner.

25. As to claims 19-21, 34 and 36-37, since the features of these claims can also be found in claims 1, 9-10, 12-14, 16-17, 22 and 33, they are rejected for the same reasons set forth in the rejection of claims 1, 9-10, 12-14, 16-17, 22 and 33 above.

26. A shortened statutory period for response to this action is set to expire 3 (three) months and 0 days from the mail date of this letter. Failure to respond within the period for response will result in ABANDONMENT of the application (see 35 U.S.C. 133, M.P.E.P. 710.02, 710.02(b)).

Conclusion

Examiner note: Examiner has cited particular columns and line numbers in the references as applied to the claims above for the convenience of the applicant. Although the specified citations are representative of the teachings of the art and are applied to the specific limitations within the individual claim, other passages and figures may apply as well. It is respectfully requested from the applicant in preparing responses, to fully consider the references in entirety as potentially teaching all or part of the claimed invention, as well as the contest of the passage as taught by the prior art or disclosed by the Examiner.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Wen-Tai Lin whose telephone number is (571)272-3969. The examiner can normally be reached on Monday-Friday (8:00-5:00) .

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John Follansbee can be reached on (571)272-3964. The fax phone numbers for the organization where this application or proceeding is assigned are as follows:

(571) 273-8300 for official communications; and

(571) 273-3969 for status inquires draft communication.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Wen-Tai Lin

July 14, 2005

Wen-Tai Lin
7/14/05